DATA SCIENCE CASE STUDY

K.ISWARYA

VU22CSEN0100284

DATASET: VIDEO GAMES SALES DATASET

The **Video Games Sales Dataset** is a comprehensive collection of data that captures the global sales, critical reviews, and user feedback for video games released across various platforms. This dataset provides valuable insights into the video game industry, allowing for analysis of trends and performance across different regions and platforms. It includes the following key attributes:

1. **Name**: The title of the video game.
2. **Platform**: The gaming console or device (e.g., PS4, Xbox One, PC).
3. **Year_of_Release**: The year in which the game was released.
4. **Genre**: The type of game (e.g., Action, Adventure, Sports).
5. **Publisher**: The company that published the game.
6. **NA_Sales**: Sales figures in North America (in millions of units).
7. **EU_Sales**: Sales figures in Europe (in millions of units).
8. **JP_Sales**: Sales figures in Japan (in millions of units).
9. **Other_Sales**: Sales figures in other regions (in millions of units).
10. **Global_Sales**: Total worldwide sales (in millions of units).
11. **Critic_Score**: The aggregated score from various critics, typically out of 100.
12. **Critic_Count**: The number of critics who reviewed the game.
13. **User_Score**: The average user score, typically on a scale from 0 to 10.
14. **User_Count**: The number of users who rated the game.
15. **Rating**: The content rating(range between 1 to 5).

Install necessary packages:

1. readr()

Description: The readr package is part of the tidyverse and provides a fast and user-friendly way to read rectangular data (like CSV files) into R. It offers functions that are optimized for performance and ease of use, helping to streamline the data import process.

Output:

```
> install.packages("readr")
Installing package into 'C:/Users/iswar/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/readr_2.1.5.zip'
Content type 'application/zip' length 1205912 bytes (1.2 MB)
downloaded 1.2 MB

package 'readr' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\iswar\AppData\Local\Temp\RtmpkdLgG9\downloaded_packages
```

2. dplyr()

Description: The dplyr package is designed for data manipulation and transformation. It provides a set of functions that facilitate data wrangling tasks, such as filtering, selecting, arranging, and summarizing data frames.

Output:

```
> install.packages("dplyr")
Installing package into 'C:/Users/iswar/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/dplyr_1.1.4.zip'
Content type 'application/zip' length 1583280 bytes (1.5 MB)
downloaded 1.5 MB

package 'dplyr' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\iswar\AppData\Local\Temp\RtmpkdLgG9\downloaded_packages
```

3. ggplot2()

Description: The ggplot2 package is a powerful visualization library based on the grammar of graphics. It enables users to create a wide variety of static and interactive plots through a coherent and layered approach.

Output:

```
> install.packages("ggplot2")
Installing package into 'C:/Users/iswar/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/ggplot2_3.5.1.zip'
Content type 'application/zip' length 5016774 bytes (4.8 MB)
downloaded 4.8 MB

package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\iswar\AppData\Local\Temp\RtmpkdLgG9\downloaded_packages
```

Exploratory Data Analysis (EDA) Functions:

1. summary()

Description: Provides a statistical summary for each column in a dataset. For numeric columns, it returns key statistics such as the minimum, 1st quartile (Q1), median, mean, 3rd quartile (Q3), and maximum. For non-numeric columns, it provides the frequency of the most common categories.

Output: We get the summary for each and every column

```
> summary(d)
     Name              Platform         Year_of_Release       Genre          Publisher          NA_Sales       EU_Sales       JP_Sales      Other_Sales      Global_Sales
 Length:299         Length:299         Length:299         Length:299         Length:299         Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.0000   Min.   : 4.04
 Class :character   Class :character   Class :character   Class :character   Class :character   1st Qu.: 2.185   1st Qu.: 1.255   1st Qu.: 0.065   1st Qu.: 0.2900   1st Qu.: 4.73
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Median : 2.990   Median : 1.950   Median : 0.280   Median : 0.5700   Median : 5.87
                                                                                              Mean   : 4.015   Mean   : 2.484   Mean   : 1.089   Mean   : 0.7724   Mean   : 8.36
                                                                                              3rd Qu.: 4.340   3rd Qu.: 2.820   3rd Qu.: 1.575   3rd Qu.: 0.8800   3rd Qu.: 8.85
                                                                                              Max.   :41.360   Max.   :28.960   Max.   :10.220   Max.   :10.5700   Max.   :82.53

 Critic_Score    Critic_Count      User_Score       User_Count      Developer            Rating
 Min.   :45.00   Min.   :  8.00   Min.   : 2.600   Min.   :   6.0   Length:299         Min.   :1.000
 1st Qu.:81.25   1st Qu.: 32.75   1st Qu.: 7.300   1st Qu.: 146.5   Class :character   1st Qu.:2.000
 Median :87.00   Median : 57.00   Median : 8.000   Median : 613.0   Mode  :character   Median :3.000
 Mean   :85.33   Mean   : 54.94   Mean   : 8.067   Mean   :1140.0                      Mean   :2.977
 3rd Qu.:92.75   3rd Qu.: 77.00   3rd Qu.: 8.600   3rd Qu.:1463.0                      3rd Qu.:4.000
 Max.   :98.00   Max.   :113.00   Max.   :86.000   Max.   :9629.0                      Max.   :5.000
 NA's   :73      NA's   :91       NA's   :88       NA's   :88
```

2. str()

Description: Displays the structure of a data frame or other R objects, showing each column's name, data type, and the first few values.

Output: The data type of every column is displayed

```
> str(d)
spc_tbl_ [299 × 16] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Name           : chr [1:299] "Wii Sports" "Super Mario Bros." "Mario Kart Wii" "Wii Sports Resort" ...
 $ Platform       : chr [1:299] "Wii" "NES" "Wii" "Wii" ...
 $ Year_of_Release: chr [1:299] "2006" "1985" "2008" "2009" ...
 $ Genre          : chr [1:299] "Sports" "Platform" "Racing" "Sports" ...
 $ Publisher      : chr [1:299] "Nintendo" "Nintendo" "Nintendo" "Nintendo" ...
 $ NA_Sales       : num [1:299] 41.4 29.1 15.7 15.6 11.3 ...
 $ EU_Sales       : num [1:299] 28.96 3.58 12.76 10.93 8.89 ...
 $ JP_Sales       : num [1:299] 3.77 6.81 3.79 3.28 10.22 ...
 $ Other_Sales    : num [1:299] 8.45 0.77 3.29 2.95 1 0.58 2.88 2.84 2.24 0.47 ...
 $ Global_Sales   : num [1:299] 82.5 40.2 35.5 32.8 31.4 ...
 $ Critic_Score   : num [1:299] 76 84 82 80 74 87 89 58 87 65 ...
 $ Critic_Count   : num [1:299] 51 75 73 73 NA NA 65 41 80 NA ...
 $ User_Score     : num [1:299] 8 86 8.3 8 NA NA 8.5 6.6 8.4 NA ...
 $ User_Count     : num [1:299] 322 600 709 192 NA NA 431 129 594 NA ...
 $ Developer      : chr [1:299] "Nintendo" NA "Nintendo" "Nintendo" ...
 $ Rating         : num [1:299] 4 4 1 5 2 5 1 3 1 2 ...
 - attr(*, "spec")=
  .. cols(
  ..   Name = col_character(),
  ..   Platform = col_character(),
  ..   Year_of_Release = col_character(),
  ..   Genre = col_character(),
  ..   Publisher = col_character(),
  ..   NA_Sales = col_double(),
  ..   EU_Sales = col_double(),
  ..   JP_Sales = col_double(),
  ..   Other_Sales = col_double(),
  ..   Global_Sales = col_double(),
  ..   Critic_Score = col_double(),
  ..   Critic_Count = col_double(),
  ..   User_Score = col_double(),
  ..   User_Count = col_double(),
  ..   Developer = col_character(),
  ..   Rating = col_double()
  .. )
 - attr(*, "problems")=<externalptr>
```

3. head()

Description: Displays the first few rows (default is 6) of the dataset. This function helps verify data loading and provides a quick glance at how the data is structured.

Output: The data of the first six rows is displayed

```
> head(d)
# A tibble: 6 × 16
  Name                  Platform Year_of_Release Genre     Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating
  <chr>                 <chr>    <chr>          <chr>     <chr>       <dbl>    <dbl>    <dbl>       <dbl>        <dbl>        <dbl>        <dbl>      <dbl>      <dbl> <chr>     <chr>
1 Wii Sports            Wii      2006           Sports    Nintendo     41.4     29.0     3.77        8.45         82.5           76           51          8        322 Nintendo  4
2 Super Mario Bros.     NES      1985           Platform  Nintendo     29.1      3.58    6.81        0.77         40.2           84           75         86        600 <NA>      4
3 Mario Kart Wii        Wii      2008           Racing    Nintendo     15.7     12.8     3.79        3.29         35.5           82           73          8.3      709 Nintendo  1
4 Wii Sports Resort     Wii      2009           Sports    Nintendo     15.6     10.9     3.28        2.95         32.8           80           73          8        192 Nintendo  5
5 Pokemon Red/Pokemon … GB       1996           Role-Pla… Nintendo     11.3      8.89   10.2         1            31.4           74           NA         NA         NA <NA>      2
6 Tetris                GB       1989           Puzzle    Nintendo     23.2      2.26    4.22        0.58         30.3           87           NA         NA         NA <NA>      5
>
```

Statistical Functions

1. mean()

Description: The mean (average) is the sum of all values divided by the number of values. It represents the central value of a dataset and is useful for understanding the overall level of a numeric variable.

Output: The mean of all the numeric columns is displayed

```
>  numeric_data <- d %>%
+ select(where(is.numeric))
>  column_means <- colMeans(numeric_data, na.rm = TRUE)
>  print(column_means)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User_Score   User_Count       Rating
   4.0147157    2.4840468    1.0891639    0.7723746    8.3601338   85.3274336   54.9423077    8.0668246 1140.0189573    2.9765886
```

2. median()

Description: The median is the middle value when the data is sorted in ascending order. If the number of values is even, the median is the average of the two middle numbers. It is a robust measure of central tendency and is not as affected by outliers as the mean.

Output: The median of all the numeric columns is displayed

```
>  column_median <- sapply(numeric_data, median, na.rm = TRUE)
> print(column_median)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User_Score   User_Count       Rating
        2.99         1.95         0.28         0.57         5.87        87.00        57.00         8.00       613.00         3.00
```

3. sd():

Description: The standard deviation measures the amount of variation or dispersion in a set of values. A low standard deviation indicates that the values are close to the mean, while a high standard deviation indicates a wider range of values.

Output: The standard deviation of all the numeric columns is displayed

```
> column_sd <- sapply(numeric_data, sd, na.rm = TRUE)
> print(column_sd)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User_Score   User_Count       Rating
    4.084114     2.468379     1.542527     1.015053     7.258410     9.809694    26.070817     5.551837 1520.193136     1.379179
```

4. **var()**

   Description: Measures how far a set of numbers is spread out from their average (mean). It is the average of the squared differences from the mean, showing data dispersion.

   Output: The variance of all the numeric columns is displayed

```
> column_variance <- sapply(numeric_data, var, na.rm = TRUE)
> print(column_variance)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User_Score   User_Count       Rating
1.667998e+01 6.092895e+00 2.379390e+00 1.030333e+00 5.268451e+01 9.623009e+01 6.796875e+02 3.082289e+01 2.310987e+06 1.902135e+00
```

5. **summary()**

   Description: Provides a five-number summary for each numeric column, including the minimum, first quartile (25th percentile), median, mean, third quartile (75th percentile), and maximum. For non-numeric data, it provides counts of unique values.

   Output: A five-number summary is displayed for each numeric column

```
> column_summary <- summary(numeric_data)
> print(column_summary)
    NA_Sales          EU_Sales          JP_Sales        Other_Sales       Global_Sales     Critic_Score    Critic_Count     User_Score      User_Count        Rating
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.0000   Min.   : 4.04   Min.   :45.00   Min.   :  8.00   Min.   : 2.600   Min.   :   6.0   Min.   :1.000
 1st Qu.: 2.185   1st Qu.: 1.255   1st Qu.: 0.065   1st Qu.: 0.2900   1st Qu.: 4.73   1st Qu.:81.25   1st Qu.: 32.75   1st Qu.: 7.300   1st Qu.: 146.5   1st Qu.:2.000
 Median : 2.990   Median : 1.950   Median : 0.280   Median : 0.5700   Median : 5.87   Median :87.00   Median : 57.00   Median : 8.000   Median : 613.0   Median :3.000
 Mean   : 4.015   Mean   : 2.484   Mean   : 1.089   Mean   : 0.7724   Mean   : 8.36   Mean   :85.33   Mean   : 54.94   Mean   : 8.067   Mean   :1140.0   Mean   :2.977
 3rd Qu.: 4.340   3rd Qu.: 2.820   3rd Qu.: 1.575   3rd Qu.: 0.8800   3rd Qu.: 8.85   3rd Qu.:92.75   3rd Qu.: 77.00   3rd Qu.: 8.600   3rd Qu.:1463.0   3rd Qu.:4.000
 Max.   :41.360   Max.   :28.960   Max.   :10.220   Max.   :10.5700   Max.   :82.53   Max.   :98.00   Max.   :113.00   Max.   :86.000   Max.   :9629.0   Max.   :5.000
                                                                                      NA's   :73      NA's   :91       NA's   :88       NA's   :88
```

6. **t.test()**

   Description: Compares the means of two groups to determine if they are significantly different from each other. A t-test assumes normal distribution and can be one-sample, two-sample (independent), or paired.

   Output: A t-test is conducted between NA_Sales and Global_Sales

```
> t_test_result <- t.test(numeric_data$NA_Sales, numeric_data$Global_Sales, paired = TRUE)
> print(t_test_result)

        Paired t-test

data:  numeric_data$NA_Sales and numeric_data$Global_Sales
t = -19.261, df = 298, p-value < 2.2e-16
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 -4.789392 -3.901444
sample estimates:
mean difference
      -4.345418
```

7. **cor()**

   Description: Measures the linear relationship between two numeric variables. The result ranges from -1 to 1:

   - 1: Perfect positive correlation
   - 0: No correlation
   - -1: Perfect negative correlation

   Output: The correlation coefficient between the columns is displayed

```
>  correlation_matrix <- cor(numeric_data, use = "pairwise.complete.obs")
> print(correlation_matrix)
              NA_Sales     EU_Sales     JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count  User_Score    User_Count       Rating
NA_Sales      1.00000000  0.619516710  0.36160761  0.45134540  0.9132756817 -0.12985205  0.119132936  0.40766622  0.0355815429 -0.035463305
EU_Sales      0.61951671  1.000000000  0.32958281  0.58073620  0.8398433037 -0.15616950  0.009478787 -0.01101245  0.0206992027 -0.050401243
JP_Sales      0.36160761  0.329582811  1.00000000  0.09506471  0.5413764175 -0.08857957 -0.010627809  0.36349764 -0.2053382109 -0.136813491
Other_Sales   0.45134540  0.580736199  0.09506471  1.00000000  0.6114790867 -0.01418010  0.112168874 -0.01397435  0.0535536051 -0.045551756
Global_Sales  0.91327568  0.839843304  0.54137642  0.61147909  1.0000000000 -0.14522422  0.081727302  0.27355775 -0.0005333643 -0.072586839
Critic_Score -0.12985205 -0.156169503 -0.08857957 -0.01418010 -0.1452242247  1.00000000  0.302852144  0.06424308  0.3306956744  0.033478789
Critic_Count  0.11913294  0.009478787 -0.01062781  0.11216887  0.0817273015  0.30285214  1.000000000  0.06192026  0.5038373955  0.003200921
User_Score    0.40766622 -0.011012448  0.36349764 -0.01397435  0.2735577463  0.06424308  0.061920258  1.00000000 -0.0750331766  0.043017064
User_Count    0.03558154  0.020699203 -0.20533821  0.05355361 -0.0005333643  0.33069567  0.503837396 -0.07503318  1.0000000000  0.093160559
Rating       -0.03546331 -0.050401243 -0.13681349 -0.04555176 -0.0725868392  0.03347879  0.003200921  0.04301706  0.0931605586  1.000000000
```

Data Manipulation Functions

1. filter()

Description: Extracts rows from a data frame that meet specified conditions. It allows for logical operations to refine data.

Output: The game names are displayed whose Year_Of_Release>2010 and Global_Sales>5

```
> filtered_data <- d %>%
+ filter(Year_of_Release > 2010, Global_Sales > 5)
> print(filtered_data)
# A tibble: 54 × 16
   Name                    Platform Year_of_Release Genre Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating
   <chr>                   <chr>    <chr>          <chr> <chr>        <dbl>    <dbl>    <dbl>       <dbl>        <dbl>        <dbl>        <dbl>      <dbl>      <dbl> <chr>      <dbl>
 1 Grand Theft Auto V      PS3      2013           Acti… Take-Two…     7.02     9.09     0.98        3.96        21.0            97           50        8.2       3994 Rockstar…      2
 2 Grand Theft Auto V      X360     2013           Acti… Take-Two…     9.66     5.14     0.06        1.41        16.3            97           58        8.1       3711 Rockstar…      2
 3 Call of Duty: Modern Warfa… X360 2011           Shoo… Activisi…     9.04     4.24     0.13        1.32        14.7            88           81        3.4       8713 Infinity…      4
 4 Call of Duty: Black Ops 3 PS4    2015           Shoo… Activisi…     6.03     5.86     0.36        2.38        14.6            68           NA         NA         NA <NA>           2
 5 Pokemon X/Pokemon Y     3DS      2013           Role… Nintendo      5.28     4.19     4.35        0.78        14.6            89           NA         NA         NA <NA>           3
 6 Call of Duty: Black Ops II PS3   2012           Shoo… Activisi…     4.99     5.73     0.65        2.42        13.8            83           21        5.3        922 Treyarch       5
 7 Call of Duty: Black Ops II X360  2012           Shoo… Activisi…     8.25     4.24     0.07        1.12        13.7            83           73        4.8       2256 Treyarch       3
 8 Call of Duty: Modern Warfa… PS3  2011           Shoo… Activisi…     5.54     5.73     0.49        1.57        13.3            88           39        3.2       5234 Infinity…      2
 9 Mario Kart 7            3DS      2011           Raci… Nintendo      5.03     4.02     2.69        0.91        12.7            85           73        8.2        632 Retro St…      1
10 Grand Theft Auto V      PS4      2014           Acti… Take-Two…     3.96     6.31     0.38        1.97        12.6            97           66        8.3       2899 Rockstar…      5
# ℹ 44 more rows
# ℹ Use `print(n = ...)` to see more rows
```

2. distinct()

Description: Returns unique rows from a data frame based on specified columns. It helps to identify and remove duplicate entries.

Output:The first output gives the distinct Names of video games.

The second output displays the distinct combination values of Platform and Genre

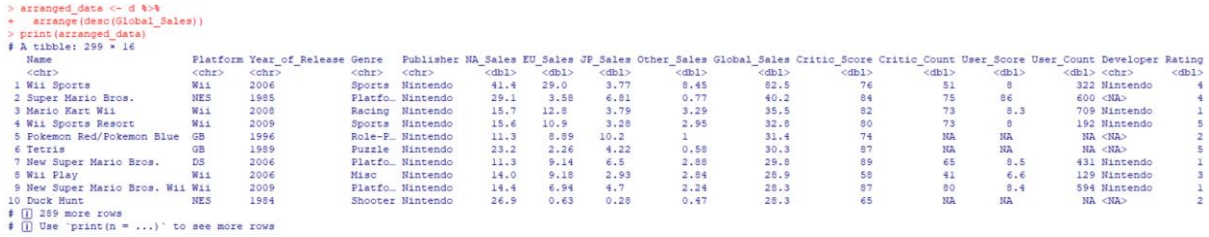
```
> distinct_titles <- d %>%
+   distinct(Name)
> print(distinct_titles)
# A tibble: 256 × 1
   Name
   <chr>
 1 Wii Sports
 2 Super Mario Bros.
 3 Mario Kart Wii
 4 Wii Sports Resort
 5 Pokemon Red/Pokemon Blue
 6 Tetris
 7 New Super Mario Bros.
 8 Wii Play
 9 New Super Mario Bros. Wii
10 Duck Hunt
# ℹ 246 more rows
# ℹ Use `print(n = ...)` to see more rows
```

```
> distinct_combinations <- d %>%
+   distinct(Platform, Genre)
> print(distinct_combinations)
# A tibble: 108 × 2
   Platform Genre
   <chr>    <chr>
 1 Wii      Sports
 2 NES      Platform
 3 Wii      Racing
 4 GB       Role-Playing
 5 GB       Puzzle
 6 DS       Platform
 7 Wii      Misc
 8 Wii      Platform
 9 NES      Shooter
10 DS       Simulation
# ℹ 98 more rows
# ℹ Use `print(n = ...)` to see more rows
```

3. **arrange()**

Description: Orders rows in a data frame based on one or more columns, either in ascending or descending order.

Output: The values of the Column Global_Sales is arranged in the descending order



```
> arranged_data <- d %>%
+   arrange(desc(Global_Sales))
> print(arranged_data)
# A tibble: 299 × 16
   Name                  Platform Year_of_Release Genre   Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating
   <chr>                 <chr>    <chr>          <chr>   <chr>       <dbl>    <dbl>    <dbl>       <dbl>        <dbl>        <dbl>        <dbl>      <dbl>      <dbl> <chr>     <chr>
 1 Wii Sports            Wii      2006           Sports  Nintendo     41.4     29.0     3.77        8.45         82.5           76           51          8        322 Nintendo  4
 2 Super Mario Bros.     NES      1985           Platfo… Nintendo     29.1      3.58    6.81        0.77         40.2           84           75         86        600 <NA>      4
 3 Mario Kart Wii        Wii      2008           Racing  Nintendo     15.7     12.8     3.79        3.29         35.5           82           73          8.3      709 Nintendo  1
 4 Wii Sports Resort     Wii      2009           Sports  Nintendo     15.6     10.9     3.28        2.95         32.8           80           73          8        192 Nintendo  5
 5 Pokemon Red/Pokemon Blue GB    1996           Role-P… Nintendo     11.3      8.89   10.2         1            31.4           74           NA         NA         NA <NA>      2
 6 Tetris                GB       1989           Puzzle  Nintendo     23.2      2.26    4.22        0.58         30.3           87           NA         NA         NA <NA>      5
 7 New Super Mario Bros. DS       2006           Platfo… Nintendo     11.3      9.14    6.5         2.88         29.8           89           65          8.5      431 Nintendo  1
 8 Wii Play              Wii      2006           Misc    Nintendo     14.0      9.18    2.93        2.84         28.9           58           41          6.6      129 Nintendo  3
 9 New Super Mario Bros. Wii Wii  2009           Platfo… Nintendo     14.4      6.94    4.7         2.24         28.3           87           80          8.4      594 Nintendo  1
10 Duck Hunt             NES      1984           Shooter Nintendo     26.9      0.63    0.28        0.47         28.3           65           NA         NA         NA <NA>      2
# ℹ 289 more rows
# ℹ Use `print(n = ...)` to see more rows
```

4. **select()**

Description: Chooses specific columns from a data frame. It can also be used to rename columns or exclude certain ones.

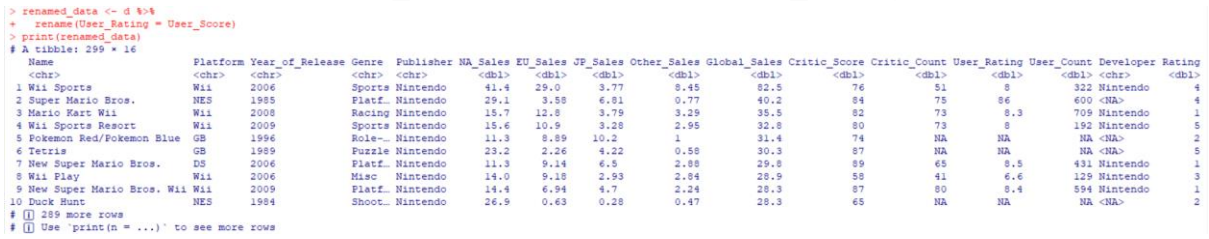
Output: Particular columns are selected like Name, Platform and Global_Sales

```
> selected_data <- d %>%
+   select(Name, Platform, Global_Sales)
> print(selected_data)
# A tibble: 299 × 3
   Name                      Platform Global_Sales
   <chr>                     <chr>           <dbl>
 1 Wii Sports                Wii              82.5
 2 Super Mario Bros.         NES              40.2
 3 Mario Kart Wii            Wii              35.5
 4 Wii Sports Resort         Wii              32.8
 5 Pokemon Red/Pokemon Blue  GB               31.4
 6 Tetris                    GB               30.3
 7 New Super Mario Bros.     DS               29.8
 8 Wii Play                  Wii              28.9
 9 New Super Mario Bros. Wii Wii              28.3
10 Duck Hunt                 NES              28.3
# ℹ 289 more rows
# ℹ Use `print(n = ...)` to see more rows
```

5. **rename()**

Description: Renames existing columns in a data frame to more meaningful or user-friendly names.

Output: The Column User_Score is renamed to User_Rating



```
> renamed_data <- d %>%
+   rename(User_Rating = User_Score)
> print(renamed_data)
# A tibble: 299 × 16
   Name                  Platform Year_of_Release Genre   Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count User_Rating User_Count Developer Rating
   <chr>                 <chr>    <chr>          <chr>   <chr>       <dbl>    <dbl>    <dbl>       <dbl>        <dbl>        <dbl>        <dbl>       <dbl>      <dbl> <chr>     <chr>
 1 Wii Sports            Wii      2006           Sports  Nintendo     41.4     29.0     3.77        8.45         82.5           76           51          8        322 Nintendo  4
 2 Super Mario Bros.     NES      1985           Platf… Nintendo     29.1      3.58    6.81        0.77         40.2           84           75         86        600 <NA>      4
 3 Mario Kart Wii        Wii      2008           Racing  Nintendo     15.7     12.8     3.79        3.29         35.5           82           73          8.3      709 Nintendo  1
 4 Wii Sports Resort     Wii      2009           Sports  Nintendo     15.6     10.9     3.28        2.95         32.8           80           73          8        192 Nintendo  5
 5 Pokemon Red/Pokemon Blue GB    1996           Role-… Nintendo     11.3      8.89   10.2         1            31.4           74           NA         NA         NA <NA>      2
 6 Tetris                GB       1989           Puzzle  Nintendo     23.2      2.26    4.22        0.58         30.3           87           NA         NA         NA <NA>      5
 7 New Super Mario Bros. DS       2006           Platf… Nintendo     11.3      9.14    6.5         2.88         29.8           89           65          8.5      431 Nintendo  1
 8 Wii Play              Wii      2006           Misc    Nintendo     14.0      9.18    2.93        2.84         28.9           58           41          6.6      129 Nintendo  3
 9 New Super Mario Bros. Wii Wii  2009           Platf… Nintendo     14.4      6.94    4.7         2.24         28.3           87           80          8.4      594 Nintendo  1
10 Duck Hunt             NES      1984           Shoot… Nintendo     26.9      0.63    0.28        0.47         28.3           65           NA         NA         NA <NA>      2
# ℹ 289 more rows
# ℹ Use `print(n = ...)` to see more rows
```

6. **mutate()**

   Description: Adds new columns or modifies existing columns in a data frame. It allows for the creation of derived variables based on calculations from existing columns.

   Output: A new column is added with name “Total_Regional_Sales”

```
> mutated_data <- d %>%
+   mutate(Total_Regional_Sales = Other_Sales + JP_Sales)
> print(mutated_data)
# A tibble: 299 × 17
   Name     Platform Year_of_Release Genre Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating Total_Regional_Sales
   <chr>    <chr>    <chr>           <chr> <chr>        <dbl>    <dbl>    <dbl>       <dbl>        <dbl>        <dbl>        <dbl> <chr>        <dbl> <chr>     <chr>                  <dbl>
 1 Wii S… Wii      2006            Spor… Nintendo      41.4     29.0     3.77        8.45         82.5           76           51 8              322 Nintendo  4                      12.2
 2 Super… NES      1985            Plat… Nintendo      29.1      3.58    6.81        0.77         40.2           84           75 86             600 <NA>      4                       7.58
 3 Mario… Wii      2008            Raci… Nintendo      15.7     12.8     3.79        3.29         35.5           82           73 8.3            709 Nintendo  1                       7.08
 4 Wii S… Wii      2009            Spor… Nintendo      15.6     10.9     3.28        2.95         32.8           80           73 8              192 Nintendo  5                       6.23
 5 Pokem… GB       1996            Role… Nintendo      11.3      8.89   10.2         1            31.4           74           NA NA              NA <NA>      2                      11.2
 6 Tetris GB       1989            Puzz… Nintendo      23.2      2.26    4.22        0.58         30.3           87           NA NA              NA <NA>      5                       4.8
 7 New S… DS       2006            Plat… Nintendo      11.3      9.14    6.5         2.88         29.8           89           65 8.5            431 Nintendo  1                       9.38
 8 Wii P… Wii      2006            Misc  Nintendo      14.0      9.18    2.93        2.84         28.9           58           41 6.6            129 Nintendo  3                       5.77
 9 New S… Wii      2009            Plat… Nintendo      14.4      6.94    4.7         2.24         28.3           87           80 8.4            594 Nintendo  1                       6.94
10 Duck … NES      1984            Shoo… Nintendo      26.9      0.63    0.28        0.47         28.3           65           NA NA              NA <NA>      2                       0.75
# ℹ 289 more rows
# ℹ Use `print(n = ...)` to see more rows
```

7. **transmute()**

   Description: Like mutate(), but it only keeps the newly created or modified columns, dropping all others.

   Output: Only the newly created column is displayed

```
> transmutated_data <- d %>%
+   transmute(Total_Regional_Sales = Other_Sales + JP_Sales)
> print(transmutated_data)
# A tibble: 299 × 1
   Total_Regional_Sales
                  <dbl>
 1                12.2
 2                 7.58
 3                 7.08
 4                 6.23
 5                11.2
 6                 4.8
 7                 9.38
 8                 5.77
 9                 6.94
10                 0.75
# ℹ 289 more rows
# ℹ Use `print(n = ...)` to see more rows
```

8. **summarize()**

   Description: Generates summary statistics for one or more columns, typically used with grouped data (using group_by()).

   Output: Summarizes the data by the condition mentioned below

```
> summary_stats <- d %>%
+   summarize(Avg_Global_Sales = mean(Global_Sales, na.rm = TRUE),
+             Median_Critic_Score = median(Critic_Score, na.rm = TRUE))
> print(summary_stats)
# A tibble: 1 × 2
  Avg_Global_Sales Median_Critic_Score
             <dbl>               <dbl>
1             8.36                  87
> 
```

Data Visualization Functions

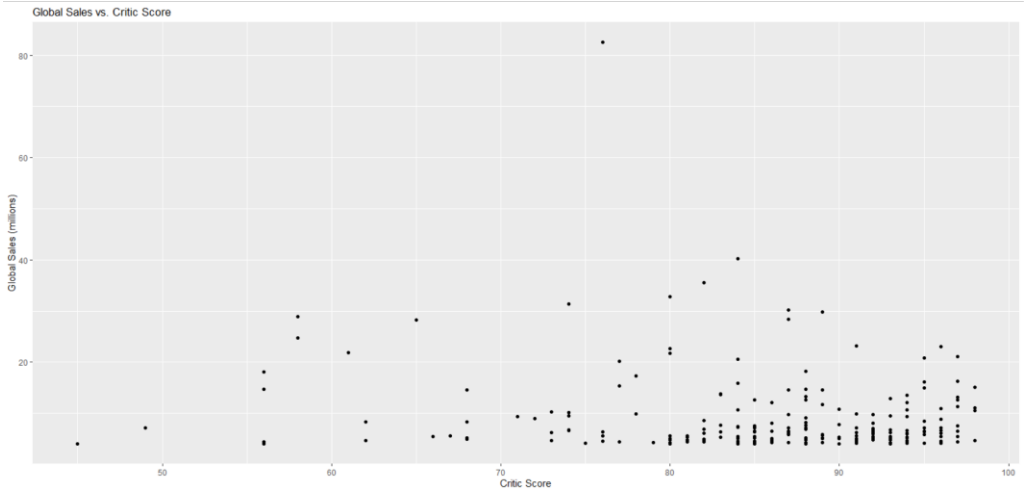
1. Scatter Plot with geom_point()

Description: Adds a scatter plot layer to a ggplot, displaying individual data points as points on the graph.

R Command:

```
> library(ggplot2)
> # Scatter plot of Global Sales vs. Critic Score
> ggplot(data = d, aes(x = Critic_Score, y = Global_Sales)) +
+   geom_point() +
+   labs(title = "Global Sales vs. Critic Score", x = "Critic Score", y = "Global Sales (millions)")
```

Output: This plot will show the relationship between the **Critic Score** and **Global Sales**:



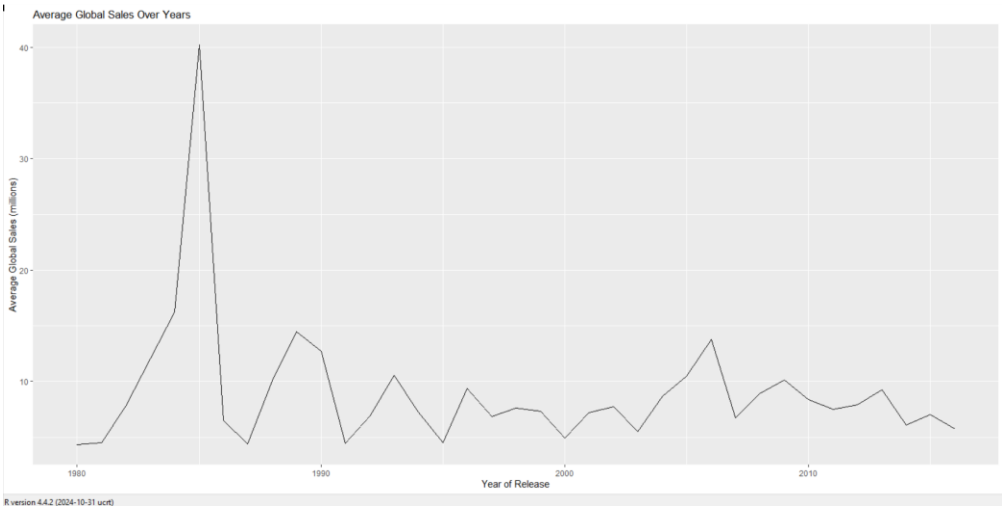
2. Line Plot with geom_line()

Description: Adds a line plot layer to a ggplot, connecting points in the order they appear in the data.

R Command:

```
> > > ggplot(data = d, aes(x = Year_of_Release, y = Global_Sales)) +
+   geom_line(stat = "summary", fun = mean) +
+   labs(title = "Average Global Sales Over Years", x = "Year of Release", y = "Average Global Sales (millions)")
```

Output: Here, we visualize the average **Global Sales** over the years of release:



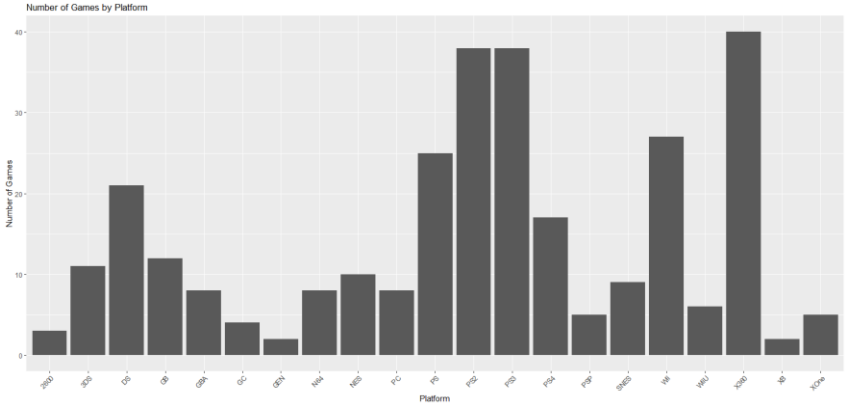
3. **Bar Plot with geom_bar()**

Description: Creates a bar chart, either counting the number of occurrences of a categorical variable or displaying values for a specific variable.

R Command:

```
> > ggplot(data = d, aes(x = Platform)) +
+   geom_bar() +
+   labs(title = "Number of Games by Platform", x = "Platform", y = "Number of Games") +
+   theme(axis.text.x = element_text(angle = 45, hjust = 1))
> ggplot(data = d, aes(x = User_Score)) +
+   geom_histogram(binwidth = 0.5, fill = "blue", color = "black") +
+   labs(title = "Distribution of User Scores", x = "User Score", y = "Count")
```

Output: This bar plot shows the number of games released on each platform:



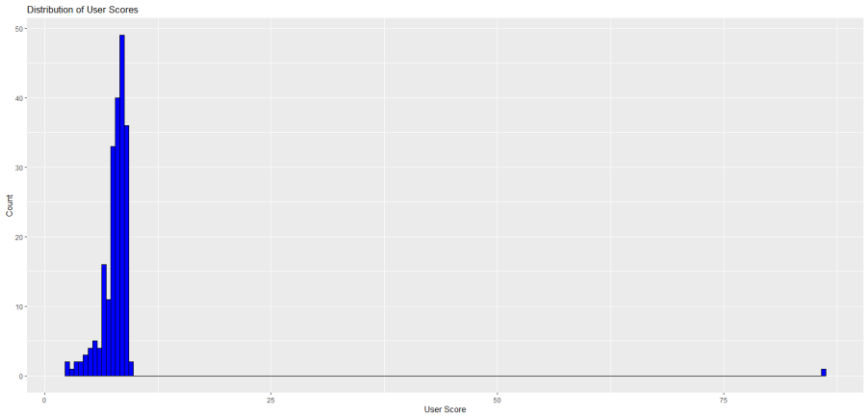
4. **Histogram with geom_histogram()**

Description: Creates a histogram, which shows the distribution of a continuous variable by dividing it into bins and counting the number of observations in each bin.

R Command:

```
> > ggplot(data = d, aes(x = Platform)) +
+   geom_bar() +
+   labs(title = "Number of Games by Platform", x = "Platform", y = "Number of Games") +
+   theme(axis.text.x = element_text(angle = 45, hjust = 1))
> ggplot(data = d, aes(x = User_Score)) +
+   geom_histogram(binwidth = 0.5, fill = "blue", color = "black") +
+   labs(title = "Distribution of User Scores", x = "User Score", y = "Count")
```

Output: Visualizing the distribution of **User Scores**:



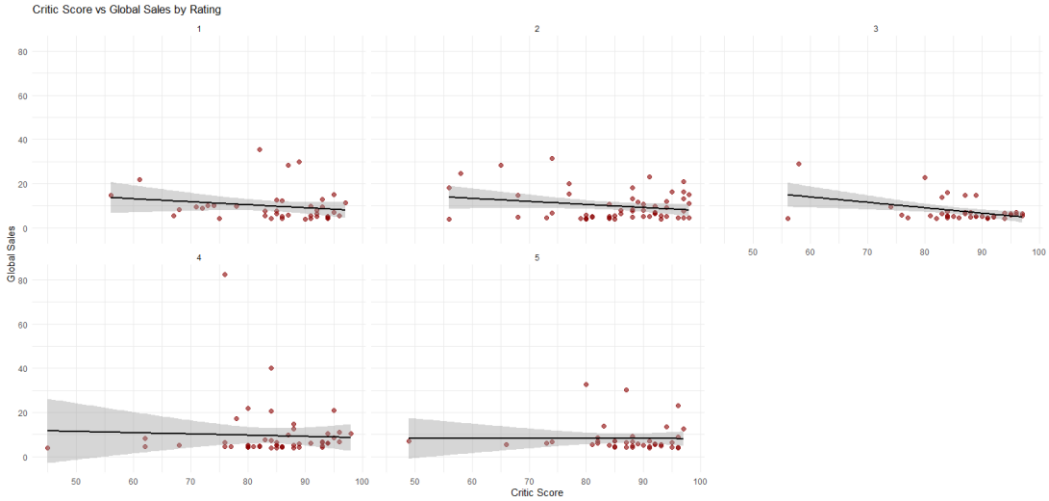
5. **Facet Wrap with facet_wrap()**

Description: Creates a series of plots (facets) based on the levels of a categorical variable, arranging them in a grid layout.

R Command:

```
> ggplot(d, aes(x = Critic_Score, y = Global_Sales)) +
+  geom_point(color = "darkred", size = 2, alpha = 0.6) +
+ geom_smooth(method = "lm", color = "black") +
+ labs(title = "Critic Score vs Global Sales by Rating", x = "Critic Score", y = "Global Sales") +
+ theme_minimal() +
+  facet_wrap(~ Rating)
```

Output: Creating separate plots for each **Rating** category showing the relationship between **Critic Score** and **Global Sales**.



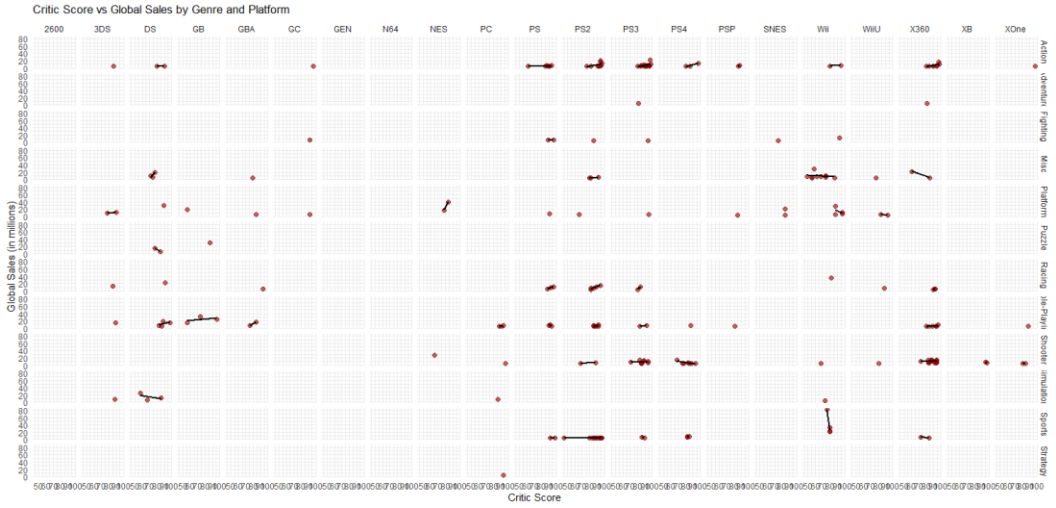
6. **Facet Grid with facet_grid()**

Description: Like facet_wrap(), but allows for creating a grid of plots based on two categorical variables.

R Command:

```
+   geom_point(color = "darkred", size = 2, alpha = 0.6) +
+   geom_smooth(method = "lm", color = "black", se = FALSE) +  # se = FALSE to remove the confidence interval
+   labs(title = "Critic Score vs Global Sales by Genre and Platform",
+        x = "Critic Score",
+        y = "Global Sales (in millions)") +
+   theme_minimal() +
+   facet_grid(Genre ~ Platform)
```

Output: This shows **Global Sales** by **Genre** and **Platform**:



```
R version 4.4.2 (2024-10-31 ucrt) -- "Pile of Leaves"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages("readr")
Installing package into ‘C:/Users/iswar/AppData/Local/R/win-library/4.4’
(as ‘lib’ is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/readr_2.1.5.zip'
Content type 'application/zip' length 1205912 bytes (1.2 MB)
downloaded 1.2 MB

package ‘readr’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\iswar\AppData\Local\Temp\RtmpkdLgG9\downloaded_packages
> install.packages("dplyr")
Installing package into ‘C:/Users/iswar/AppData/Local/R/win-library/4.4’
(as ‘lib’ is unspecified)
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/dplyr_1.1.4.zip'
Content type 'application/zip' length 1583280 bytes (1.5 MB)
downloaded 1.5 MB

package ‘dplyr’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\iswar\AppData\Local\Temp\RtmpkdLgG9\downloaded_packages
> library(readr)
> d<-read_csv("C:\\Users\\iswar\\Desktop\\R\\videogames_dataset.csv")
[1mindexing[0m [34mvideogames_dataset.csv[0m [=========================================
================================================================================] [32m488
.48MB/s[0m, eta: [36m 0s[0m

                                                        Rows: 299 Columns: 16
── Column specification ────────────────────────────────────────────────────────
────────────────────────────────────────────────────────────────────────────────
Delimiter: ","
chr  (6): Name, Platform, Year_of_Release, Genre, Publisher, Developer
dbl (10): NA_Sales, EU_Sales, JP_Sales, Other_Sales, Global_Sales, Critic_Score, Critic_Count, Us
er_Score, User_Count, Rating

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
> spec(d)
cols(
  Name = col_character(),
  Platform = col_character(),
  Year_of_Release = col_character(),
  Genre = col_character(),
  Publisher = col_character(),
  NA_Sales = col_double(),
  EU_Sales = col_double(),
  JP_Sales = col_double(),
  Other_Sales = col_double(),
  Global_Sales = col_double(),
```

```
  Critic_Score = col_double(),
  Critic_Count = col_double(),
  User_Score = col_double(),
  User_Count = col_double(),
  Developer = col_character(),
  Rating = col_double()
)
> summary(d)
     Name              Platform         Year_of_Release       Genre             Publisher
  NA_Sales         EU_Sales         JP_Sales       Other_Sales       Global_Sales
 Length:299         Length:299         Length:299         Length:299         Length:299         M
in.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.0000   Min.   : 4.04
 Class :character   Class :character   Class :character   Class :character   Class :character   1
st Qu.: 2.185   1st Qu.: 1.255   1st Qu.: 0.065   1st Qu.: 0.2900   1st Qu.: 4.73
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   M
edian : 2.990   Median : 1.950   Median : 0.280   Median : 0.5700   Median : 5.87
                                                                                               M
ean   : 4.015   Mean   : 2.484   Mean   : 1.089   Mean   : 0.7724   Mean   : 8.36
                                                                                               3
rd Qu.: 4.340   3rd Qu.: 2.820   3rd Qu.: 1.575   3rd Qu.: 0.8800   3rd Qu.: 8.85
                                                                                               M
ax.   :41.360   Max.   :28.960   Max.   :10.220   Max.   :10.5700   Max.   :82.53


  Critic_Score    Critic_Count      User_Score       User_Count      Developer             Rating

 Min.   :45.00   Min.   :  8.00   Min.   : 2.600   Min.   :   6.0   Length:299         Min.   :1.
000
 1st Qu.:81.25   1st Qu.: 32.75   1st Qu.: 7.300   1st Qu.: 146.5   Class :character   1st Qu.:2.
000
 Median :87.00   Median : 57.00   Median : 8.000   Median : 613.0   Mode  :character   Median :3.
000
 Mean   :85.33   Mean   : 54.94   Mean   : 8.067   Mean   :1140.0                      Mean   :2.
977
 3rd Qu.:92.75   3rd Qu.: 77.00   3rd Qu.: 8.600   3rd Qu.:1463.0                      3rd Qu.:4.
000
 Max.   :98.00   Max.   :113.00   Max.   :86.000   Max.   :9629.0                      Max.   :5.
000
 NA's   :73      NA's   :91       NA's   :88       NA's   :88

> str(d)
spc_tbl_ [299 × 16] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Name           : chr [1:299] "Wii Sports" "Super Mario Bros." "Mario Kart Wii" "Wii Sports Res
ort" ...
 $ Platform       : chr [1:299] "Wii" "NES" "Wii" "Wii" ...
 $ Year_of_Release: chr [1:299] "2006" "1985" "2008" "2009" ...
 $ Genre          : chr [1:299] "Sports" "Platform" "Racing" "Sports" ...
 $ Publisher      : chr [1:299] "Nintendo" "Nintendo" "Nintendo" "Nintendo" ...
 $ NA_Sales       : num [1:299] 41.4 29.1 15.7 15.6 11.3 ...
 $ EU_Sales       : num [1:299] 28.96 3.58 12.76 10.93 8.89 ...
 $ JP_Sales       : num [1:299] 3.77 6.81 3.79 3.28 10.22 ...
 $ Other_Sales    : num [1:299] 8.45 0.77 3.29 2.95 1 0.58 2.88 2.84 2.24 0.47 ...
 $ Global_Sales   : num [1:299] 82.5 40.2 35.5 32.8 31.4 ...
 $ Critic_Score   : num [1:299] 76 84 82 80 74 87 89 58 87 65 ...
 $ Critic_Count   : num [1:299] 51 75 73 73 NA NA 65 41 80 NA ...
 $ User_Score     : num [1:299] 8 86 8.3 8 NA NA 8.5 6.6 8.4 NA ...
 $ User_Count     : num [1:299] 322 600 709 192 NA NA 431 129 594 NA ...
 $ Developer      : chr [1:299] "Nintendo" NA "Nintendo" "Nintendo" ...
 $ Rating         : num [1:299] 4 4 1 5 2 5 1 3 1 2 ...
 - attr(*, "spec")=
  .. cols(
  ..   Name = col_character(),
  ..   Platform = col_character(),
  ..   Year_of_Release = col_character(),
  ..   Genre = col_character(),
  ..   Publisher = col_character(),
  ..   NA_Sales = col_double(),
  ..   EU_Sales = col_double(),
  ..   JP_Sales = col_double(),
  ..   Other_Sales = col_double(),
```

```
  ..   Global_Sales = col_double(),
  ..   Critic_Score = col_double(),
  ..   Critic_Count = col_double(),
  ..   User_Score = col_double(),
  ..   User_Count = col_double(),
  ..   Developer = col_character(),
  ..   Rating = col_double()
  .. )
 - attr(*, "problems")=<externalptr>
> head(d)
# A tibble: 6 × 16
  Name                     Platform Year_of_Release Genre     Publisher NA_Sales EU_Sales JP_Sale
s Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating
  <chr>                    <chr>    <chr>          <chr>     <chr>        <dbl>    <dbl>    <dbl
>       <dbl>        <dbl>        <dbl>        <dbl>      <dbl>      <dbl> <chr>      <dbl>
1 Wii Sports               Wii      2006           Sports    Nintendo      41.4    29.0      3.7
7        8.45         82.5           76           51        8           322 Nintendo       4
2 Super Mario Bros.        NES      1985           Platform  Nintendo      29.1     3.58     6.8
1        0.77         40.2           84           75       86           600 <NA>           4
3 Mario Kart Wii           Wii      2008           Racing    Nintendo      15.7    12.8      3.7
9        3.29         35.5           82           73        8.3         709 Nintendo       1
4 Wii Sports Resort        Wii      2009           Sports    Nintendo      15.6    10.9      3.2
8        2.95         32.8           80           73        8           192 Nintendo       5
5 Pokemon Red/Pokemon Blue GB       1996           Role-Pla… Nintendo      11.3     8.89    10.2
         1            31.4           74           NA       NA            NA <NA>           2
6 Tetris                   GB       1989           Puzzle    Nintendo      23.2     2.26     4.2
2        0.58         30.3           87           NA       NA            NA <NA>           5
>  numeric_data <- d %>%
+ select(where(is.numeric))
Error in d %>% select(where(is.numeric)) : could not find function "%>%"
> library(dplyr)

Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union

>  numeric_data <- d %>%
+ select(where(is.numeric))
>  column_means <- colMeans(numeric_data, na.rm = TRUE)
>  print(column_means)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User
_Score   User_Count       Rating
   4.0147157    2.4840468    1.0891639    0.7723746    8.3601338   85.3274336   54.9423077    8.0
668246 1140.0189573    2.9765886
>  column_median <- sapply(numeric_data, median, na.rm = TRUE)
> print(column_median)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User
_Score   User_Count       Rating
        2.99         1.95         0.28         0.57         5.87        87.00        57.00
  8.00       613.00         3.00
> column_sd <- sapply(numeric_data, sd, na.rm = TRUE)
> print(column_sd)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User
_Score   User_Count       Rating
    4.084114     2.468379     1.542527     1.015053     7.258410     9.809694    26.070817     5.
551837  1520.193136     1.379179
> column_variance <- sapply(numeric_data, var, na.rm = TRUE)
> print(column_variance)
    NA_Sales     EU_Sales     JP_Sales  Other_Sales Global_Sales Critic_Score Critic_Count   User
_Score   User_Count       Rating
1.667998e+01 6.092895e+00 2.379390e+00 1.030333e+00 5.268451e+01 9.623009e+01 6.796875e+02 3.0822
89e+01 2.310987e+06 1.902135e+00
> column_summary <- summary(numeric_data)
> print(column_summary)
```

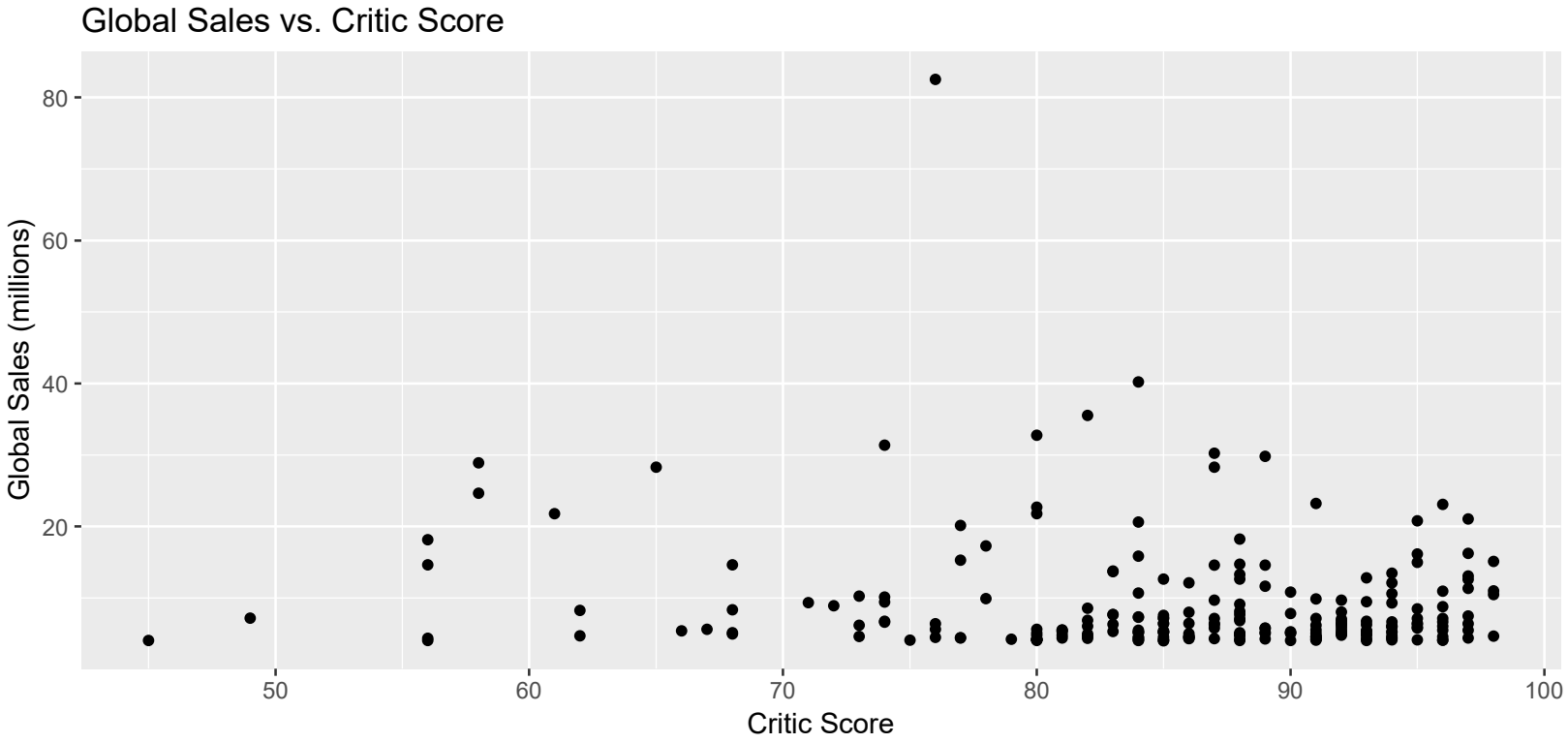
```
    NA_Sales         EU_Sales         JP_Sales       Other_Sales       Global_Sales    Critic_Sco
re    Critic_Count      User_Score       User_Count         Rating
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.0000   Min.   : 4.04   Min.   :45.
00   Min.   :  8.00   Min.   : 2.600   Min.   :   6.0   Min.   :1.000
 1st Qu.: 2.185   1st Qu.: 1.255   1st Qu.: 0.065   1st Qu.: 0.2900   1st Qu.: 4.73   1st Qu.:81.
25   1st Qu.: 32.75   1st Qu.: 7.300   1st Qu.: 146.5   1st Qu.:2.000
 Median : 2.990   Median : 1.950   Median : 0.280   Median : 0.5700   Median : 5.87   Median :87.
00   Median : 57.00   Median : 8.000   Median : 613.0   Median :3.000
 Mean   : 4.015   Mean   : 2.484   Mean   : 1.089   Mean   : 0.7724   Mean   : 8.36   Mean   :85.
33   Mean   : 54.94   Mean   : 8.067   Mean   :1140.0   Mean   :2.977
 3rd Qu.: 4.340   3rd Qu.: 2.820   3rd Qu.: 1.575   3rd Qu.: 0.8800   3rd Qu.: 8.85   3rd Qu.:92.
75   3rd Qu.: 77.00   3rd Qu.: 8.600   3rd Qu.:1463.0   3rd Qu.:4.000
 Max.   :41.360   Max.   :28.960   Max.   :10.220   Max.   :10.5700   Max.   :82.53   Max.   :98.
00   Max.   :113.00   Max.   :86.000   Max.   :9629.0   Max.   :5.000
                                                                                    NA's   :73
     NA's   :91       NA's   :88       NA's   :88
> t_test_result <- t.test(numeric_data$NA_Sales, numeric_data$Global_Sales, paired = TRUE)
> print(t_test_result)

        Paired t-test

data:  numeric_data$NA_Sales and numeric_data$Global_Sales
t = -19.261, df = 298, p-value < 2.2e-16
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 -4.789392 -3.901444
sample estimates:
mean difference
      -4.345418

>  correlation_matrix <- cor(numeric_columns, use = "pairwise.complete.obs")
Error: object 'numeric_columns' not found
>  correlation_matrix <- cor(numeric_data, use = "pairwise.complete.obs")
> print(correlation_matrix)
                NA_Sales     EU_Sales    JP_Sales Other_Sales  Global_Sales Critic_Score Critic_C
ount  User_Score    User_Count       Rating
NA_Sales      1.00000000  0.619516710  0.36160761  0.45134540  0.9132756817  -0.12985205  0.11913
2936  0.40766622  0.0355815429 -0.035463305
EU_Sales      0.61951671  1.000000000  0.32958281  0.58073620  0.8398433037  -0.15616950  0.00947
8787 -0.01101245  0.0206992027 -0.050401243
JP_Sales      0.36160761  0.329582811  1.00000000  0.09506471  0.5413764175  -0.08857957 -0.01062
7809  0.36349764 -0.2053382109 -0.136813491
Other_Sales   0.45134540  0.580736199  0.09506471  1.00000000  0.6114790867  -0.01418010  0.11216
8874 -0.01397435  0.0535536051 -0.045551756
Global_Sales  0.91327568  0.839843304  0.54137642  0.61147909  1.0000000000  -0.14522422  0.08172
7302  0.27355775 -0.0005333643 -0.072586839
Critic_Score -0.12985205 -0.156169503 -0.08857957 -0.01418010 -0.1452242247   1.00000000  0.30285
2144  0.06424308  0.3306956744  0.033478789
Critic_Count  0.11913294  0.009478787 -0.01062781  0.11216887  0.0817273015   0.30285214  1.00000
0000  0.06192026  0.5038373955  0.003200921
User_Score    0.40766622 -0.011012448  0.36349764 -0.01397435  0.2735577463   0.06424308  0.06192
0258  1.00000000 -0.0750331766  0.043017064
User_Count    0.03558154  0.020699203 -0.20533821  0.05355361 -0.0005333643   0.33069567  0.50383
7396 -0.07503318  1.0000000000  0.093160559
Rating       -0.03546331 -0.050401243 -0.13681349 -0.04555176 -0.0725868392   0.03347879  0.00320
0921  0.04301706  0.0931605586  1.000000000
> filtered_data <- d %>%
+ filter(Year_of_Release > 2010, Global_Sales > 5)
> print(filtered_data)
# A tibble: 54 × 16
   Name                        Platform Year_of_Release Genre Publisher NA_Sales EU_Sales JP_Sale
s Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating
   <chr>                       <chr>    <chr>          <chr> <chr>        <dbl>    <dbl>    <dbl
>       <dbl>        <dbl>        <dbl>        <dbl>      <dbl>      <dbl> <chr>      <dbl>
 1 Grand Theft Auto V          PS3      2013           Acti… Take-Two…     7.02     9.09     0.9
8        3.96         21.0           97           50        8.2       3994 Rockstar…       2
 2 Grand Theft Auto V          X360     2013           Acti… Take-Two…     9.66     5.14     0.0
6        1.41         16.3           97           58        8.1       3711 Rockstar…       2
 3 Call of Duty: Modern Warfa… X360     2011           Shoo… Activisi…     9.04     4.24     0.1
3        1.32         14.7           88           81        3.4       8713 Infinity…       4
```

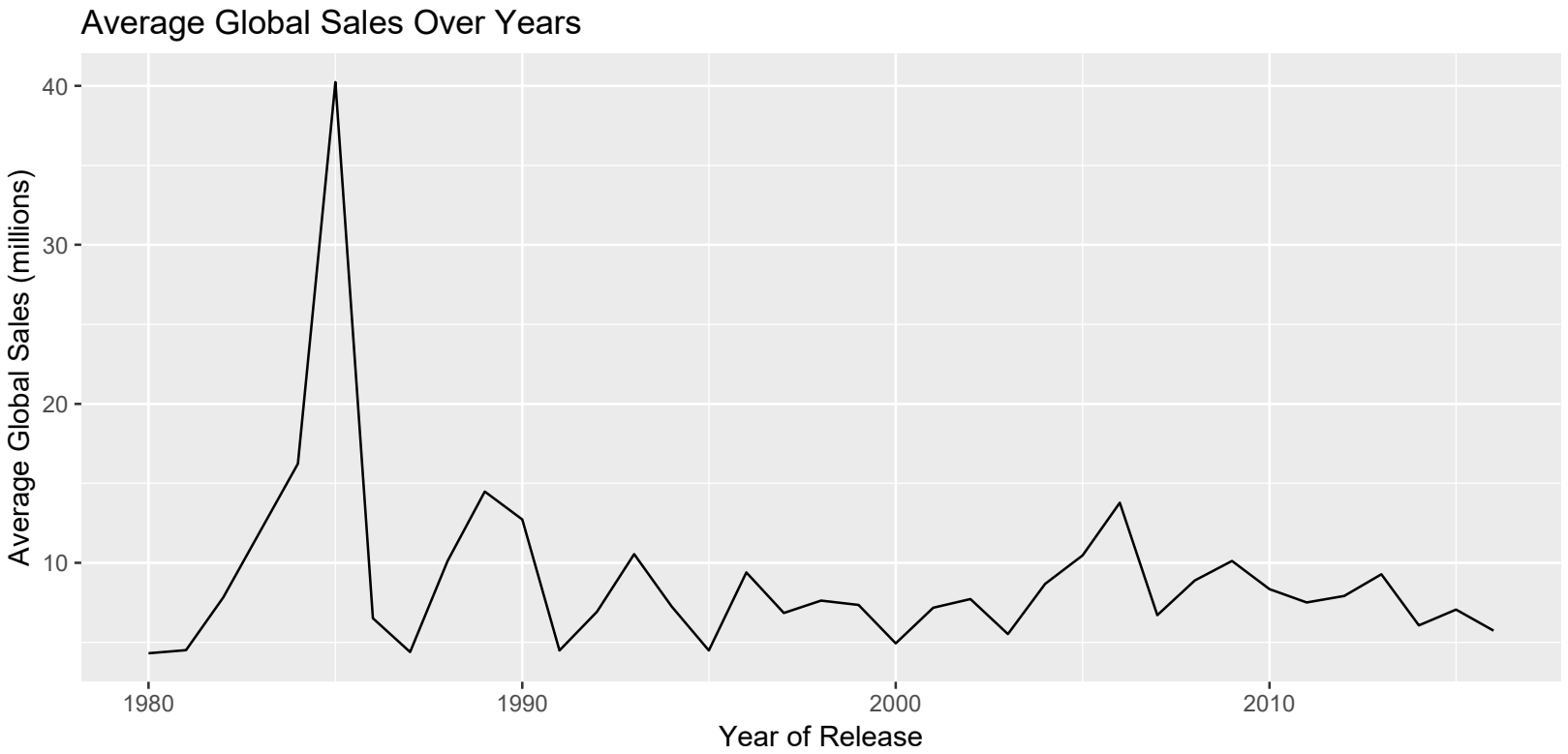
```
 4 Call of Duty: Black Ops 3   PS4      2015            Shoo… Activisi…     6.03     5.86     0.3
6        2.38         14.6           68           NA       NA           NA <NA>           2
 5 Pokemon X/Pokemon Y         3DS      2013            Role… Nintendo      5.28     4.19     4.3
5        0.78         14.6           89           NA       NA           NA <NA>           3
 6 Call of Duty: Black Ops II  PS3      2012            Shoo… Activisi…     4.99     5.73     0.6
5        2.42         13.8           83           21        5.3        922 Treyarch       5
 7 Call of Duty: Black Ops II  X360     2012            Shoo… Activisi…     8.25     4.24     0.0
7        1.12         13.7           83           73        4.8       2256 Treyarch       3
 8 Call of Duty: Modern Warfa… PS3      2011            Shoo… Activisi…     5.54     5.73     0.4
9        1.57         13.3           88           39        3.2       5234 Infinity…      2
 9 Mario Kart 7                3DS      2011            Raci… Nintendo      5.03     4.02     2.6
9        0.91         12.7           85           73        8.2        632 Retro St…      1
10 Grand Theft Auto V          PS4      2014            Acti… Take-Two…     3.96     6.31     0.3
8        1.97         12.6           97           66        8.3       2899 Rockstar…      5
# ⓘ 44 more rows
# ⓘ Use `print(n = ...)` to see more rows
> distinct_titles <- d %>%
+   distinct(Name)
> print(distinct_titles)
# A tibble: 256 × 1
   Name
   <chr>
 1 Wii Sports
 2 Super Mario Bros.
 3 Mario Kart Wii
 4 Wii Sports Resort
 5 Pokemon Red/Pokemon Blue
 6 Tetris
 7 New Super Mario Bros.
 8 Wii Play
 9 New Super Mario Bros. Wii
10 Duck Hunt
# ⓘ 246 more rows
# ⓘ Use `print(n = ...)` to see more rows
> distinct_combinations <- d %>%
+   distinct(Platform, Genre)
> print(distinct_combinations)
# A tibble: 108 × 2
   Platform Genre
   <chr>    <chr>
 1 Wii      Sports
 2 NES      Platform
 3 Wii      Racing
 4 GB       Role-Playing
 5 GB       Puzzle
 6 DS       Platform
 7 Wii      Misc
 8 Wii      Platform
 9 NES      Shooter
10 DS       Simulation
# ⓘ 98 more rows
# ⓘ Use `print(n = ...)` to see more rows
> arranged_data <- d %>%
+   arrange(desc(Global_Sales))
> print(arranged_data)
# A tibble: 299 × 16
   Name                     Platform Year_of_Release Genre   Publisher NA_Sales EU_Sales JP_Sale
s Other_Sales Global_Sales Critic_Score Critic_Count User_Score User_Count Developer Rating
   <chr>                    <chr>    <chr>          <chr>   <chr>       <dbl>    <dbl>    <dbl
>       <dbl>        <dbl>        <dbl>        <dbl>      <dbl> <chr>      <dbl>
 1 Wii Sports               Wii      2006           Sports  Nintendo     41.4     29.0      3.7
7        8.45         82.5           76           51        8          322 Nintendo       4
 2 Super Mario Bros.        NES      1985           Platfo… Nintendo     29.1      3.58     6.8
1        0.77         40.2           84           75       86          600 <NA>           4
 3 Mario Kart Wii           Wii      2008           Racing  Nintendo     15.7     12.8      3.7
9        3.29         35.5           82           73        8.3        709 Nintendo       1
 4 Wii Sports Resort        Wii      2009           Sports  Nintendo     15.6     10.9      3.2
8        2.95         32.8           80           73        8          192 Nintendo       5
 5 Pokemon Red/Pokemon Blue GB       1996           Role-P… Nintendo     11.3      8.89    10.2
         1            31.4           74           NA       NA           NA <NA>           2
```

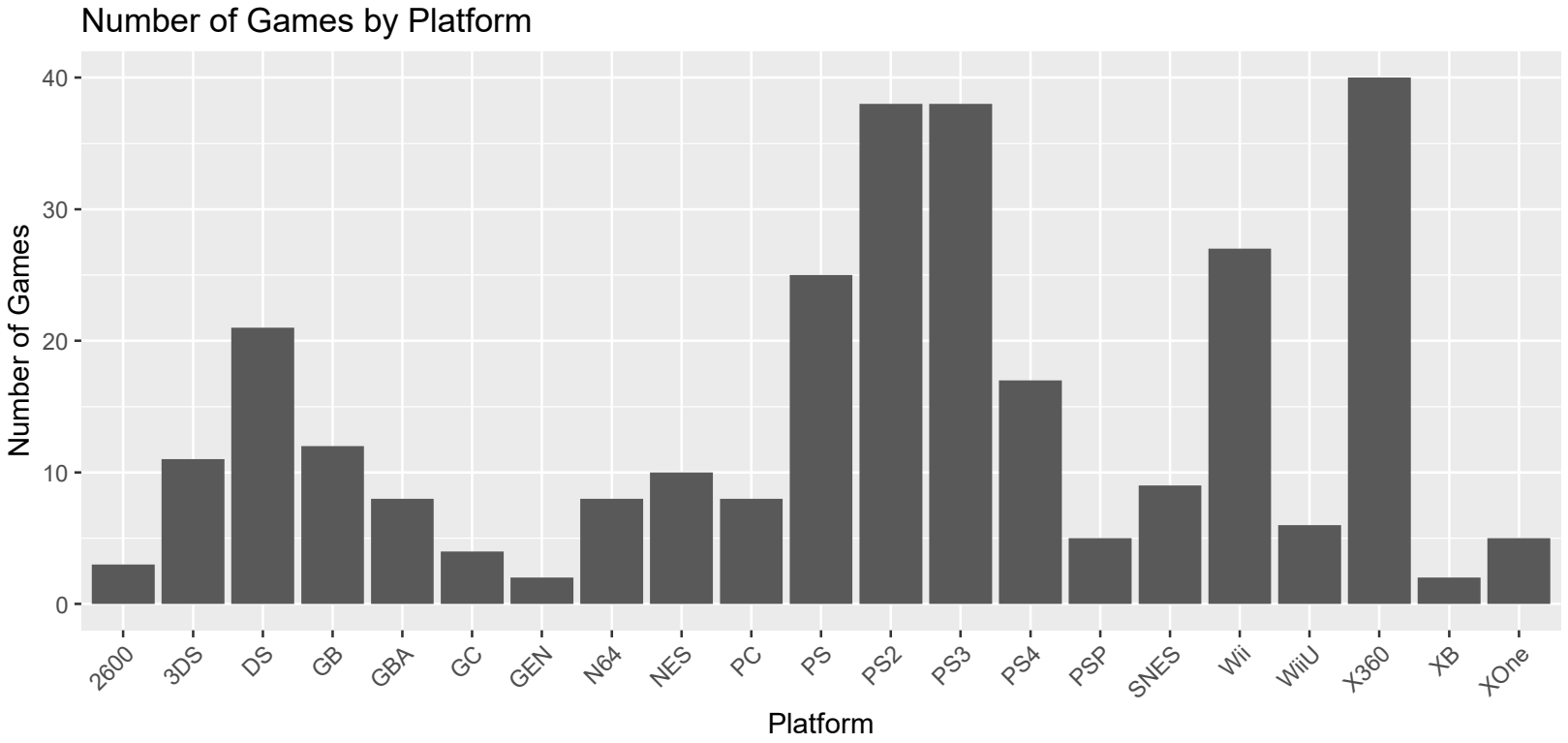
```
 6 Tetris                    GB       1989           Puzzle  Nintendo      23.2     2.26     4.2
2        0.58         30.3          87          NA       NA           NA <NA>           5
 7 New Super Mario Bros.     DS       2006           Platfo… Nintendo      11.3     9.14     6.5
         2.88         29.8          89          65        8.5        431 Nintendo       1
 8 Wii Play                  Wii      2006           Misc    Nintendo      14.0     9.18     2.9
3        2.84         28.9          58          41        6.6        129 Nintendo       3
 9 New Super Mario Bros. Wii Wii      2009           Platfo… Nintendo      14.4     6.94     4.7
         2.24         28.3          87          80        8.4        594 Nintendo       1
10 Duck Hunt                 NES      1984           Shooter Nintendo      26.9     0.63     0.2
8        0.47         28.3          65          NA       NA           NA <NA>           2
# ⓘ 289 more rows
# ⓘ Use `print(n = ...)` to see more rows
> selected_data <- d %>%
+   select(Name, Platform, Global_Sales)
> print(selected_data)
# A tibble: 299 × 3
   Name                      Platform Global_Sales
   <chr>                     <chr>           <dbl>
 1 Wii Sports                Wii              82.5
 2 Super Mario Bros.         NES              40.2
 3 Mario Kart Wii            Wii              35.5
 4 Wii Sports Resort         Wii              32.8
 5 Pokemon Red/Pokemon Blue  GB               31.4
 6 Tetris                    GB               30.3
 7 New Super Mario Bros.     DS               29.8
 8 Wii Play                  Wii              28.9
 9 New Super Mario Bros. Wii Wii              28.3
10 Duck Hunt                 NES              28.3
# ⓘ 289 more rows
# ⓘ Use `print(n = ...)` to see more rows
> renamed_data <- d %>%
+   rename(User_Rating = User_Score)
> print(renamed_data)
# A tibble: 299 × 16
   Name                      Platform Year_of_Release Genre  Publisher NA_Sales EU_Sales JP_Sales
 Other_Sales Global_Sales Critic_Score Critic_Count User_Rating User_Count Developer Rating
   <chr>                     <chr>    <chr>          <chr>  <chr>  <chr>       <dbl>    <dbl>    <dbl>
       <dbl>        <dbl>        <dbl>        <dbl>          <dbl>      <dbl> <chr>      <dbl>
 1 Wii Sports                Wii      2006           Sports Nintendo      41.4    29.0      3.77
       8.45         82.5          76          51        8          322 Nintendo       4
 2 Super Mario Bros.         NES      1985           Platf… Nintendo      29.1     3.58     6.81
       0.77         40.2          84          75        86         600 <NA>           4
 3 Mario Kart Wii            Wii      2008           Racing Nintendo      15.7    12.8      3.79
       3.29         35.5          82          73        8.3        709 Nintendo       1
 4 Wii Sports Resort         Wii      2009           Sports Nintendo      15.6    10.9      3.28
       2.95         32.8          80          73        8          192 Nintendo       5
 5 Pokemon Red/Pokemon Blue  GB       1996           Role-… Nintendo      11.3     8.89    10.2
       1            31.4          74          NA       NA           NA <NA>           2
 6 Tetris                    GB       1989           Puzzle Nintendo      23.2     2.26     4.22
       0.58         30.3          87          NA       NA           NA <NA>           5
 7 New Super Mario Bros.     DS       2006           Platf… Nintendo      11.3     9.14     6.5
       2.88         29.8          89          65        8.5        431 Nintendo       1
 8 Wii Play                  Wii      2006           Misc   Nintendo      14.0     9.18     2.93
       2.84         28.9          58          41        6.6        129 Nintendo       3
 9 New Super Mario Bros. Wii Wii      2009           Platf… Nintendo      14.4     6.94     4.7
       2.24         28.3          87          80        8.4        594 Nintendo       1
10 Duck Hunt                 NES      1984           Shoot… Nintendo      26.9     0.63     0.28
       0.47         28.3          65          NA       NA           NA <NA>           2
# ⓘ 289 more rows
# ⓘ Use `print(n = ...)` to see more rows
> mutated_data <- d %>%
+   mutate(Total_Regional_Sales = Other_Sales + JP_Sales)
> print(mutated_data)
# A tibble: 299 × 17
   Name   Platform Year_of_Release Genre Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_
Sales Critic_Score Critic_Count User_Score User_Count Developer Rating Total_Regional_Sales
   <chr>  <chr>    <chr>          <chr> <chr>      <dbl>    <dbl>    <dbl>       <dbl>
<dbl>        <dbl>        <dbl>      <dbl>      <dbl> <chr>      <dbl>                <dbl>
 1 Wii S… Wii      2006           Spor… Nintendo      41.4    29.0      3.77        8.45
 82.5           76           51         8          322 Nintendo       4                12.2
```

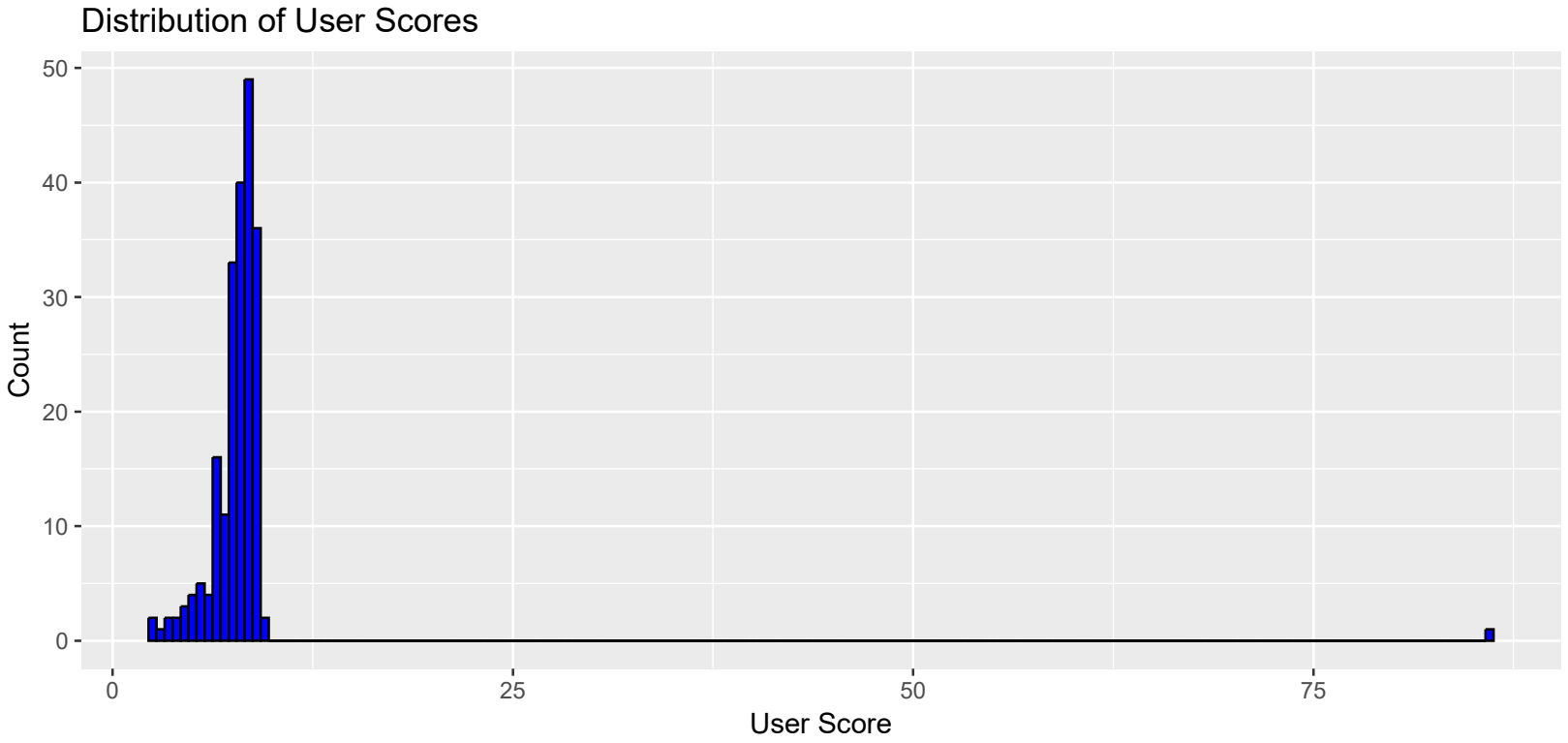
```
 2 Super… NES      1985            Plat… Nintendo      29.1     3.58     6.81        0.77
 40.2           84           75       86          600 <NA>            4                  7.58
 3 Mario… Wii      2008            Raci… Nintendo      15.7    12.8      3.79        3.29
 35.5           82           73        8.3        709 Nintendo        1                  7.08
 4 Wii S… Wii      2009            Spor… Nintendo      15.6    10.9      3.28        2.95
 32.8           80           73        8          192 Nintendo        5                  6.23
 5 Pokem… GB       1996            Role… Nintendo      11.3     8.89    10.2         1
 31.4           74           NA       NA           NA <NA>            2                 11.2
 6 Tetris GB       1989            Puzz… Nintendo      23.2     2.26     4.22        0.58
 30.3           87           NA       NA           NA <NA>            5                  4.8
 7 New S… DS       2006            Plat… Nintendo      11.3     9.14     6.5         2.88
 29.8           89           65        8.5        431 Nintendo        1                  9.38
 8 Wii P… Wii      2006            Misc  Nintendo      14.0     9.18     2.93        2.84
 28.9           58           41        6.6        129 Nintendo        3                  5.77
 9 New S… Wii      2009            Plat… Nintendo      14.4     6.94     4.7         2.24
 28.3           87           80        8.4        594 Nintendo        1                  6.94
10 Duck … NES      1984            Shoo… Nintendo      26.9     0.63     0.28        0.47
 28.3           65           NA       NA           NA <NA>            2                  0.75
# ⓘ 289 more rows
# ⓘ Use `print(n = ...)` to see more rows
> transmutated_data <- d %>%
+   transmute(Total_Regional_Sales = Other_Sales + JP_Sales)
> print(transmutated_data)
# A tibble: 299 × 1
   Total_Regional_Sales
                  <dbl>
 1                 12.2
 2                  7.58
 3                  7.08
 4                  6.23
 5                 11.2
 6                  4.8
 7                  9.38
 8                  5.77
 9                  6.94
10                  0.75
# ⓘ 289 more rows
# ⓘ Use `print(n = ...)` to see more rows
> summary_stats <- d %>%
+   summarize(Avg_Global_Sales = mean(Global_Sales, na.rm = TRUE),
+             Median_Critic_Score = median(Critic_Score, na.rm = TRUE))
> print(summary_stats)
# A tibble: 1 × 2
  Avg_Global_Sales Median_Critic_Score
             <dbl>               <dbl>
1             8.36                  87
> install.packages("ggplot2")
Installing package into 'C:/Users/iswar/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
trying URL 'https://cran.icts.res.in/bin/windows/contrib/4.4/ggplot2_3.5.1.zip'
Content type 'application/zip' length 5016774 bytes (4.8 MB)
downloaded 4.8 MB

package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\iswar\AppData\Local\Temp\RtmpkdLgG9\downloaded_packages
> library(ggplot2)
> # Scatter plot of Global Sales vs. Critic Score
> ggplot(data = d, aes(x = Critic_Score, y = Global_Sales)) +
+   geom_point() +
+   labs(title = "Global Sales vs. Critic Score", x = "Critic Score", y = "Global Sales (millions
)")
Warning message:
Removed 73 rows containing missing values or values outside the scale range
(`geom_point()`).
Error in UseMethod("depth") :
  no applicable method for 'depth' applied to an object of class "NULL"
Error in UseMethod("depth") :
  no applicable method for 'depth' applied to an object of class "NULL"
```

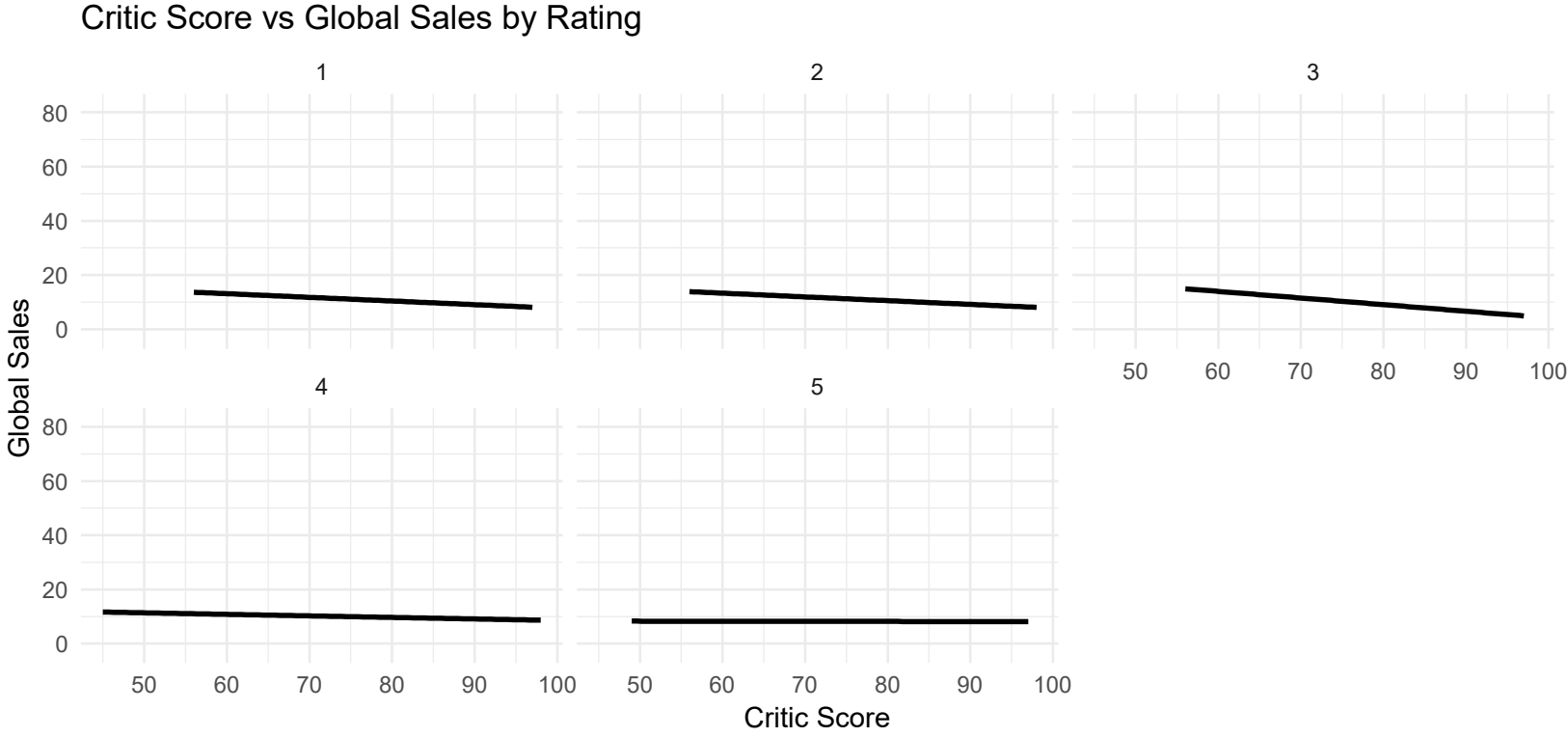
```
> > > ggplot(data = d, aes(x = Year_of_Release, y = Global_Sales)) +
+   geom_line(stat = "summary", fun = mean) +
+   labs(title = "Average Global Sales Over Years", x = "Year of Release", y = "Average Global Sa
les (millions)")
`geom_line()`: Each group consists of only one observation.
ⓘ Do you need to adjust the group aesthetic?
Error in grid.Call.graphics(C_upviewport, as.integer(n)) :
  cannot pop the top-level viewport ('grid' and 'graphics' output mixed?)
> > ggplot(data = d, aes(x = Platform)) +
+   geom_bar() +
+   labs(title = "Number of Games by Platform", x = "Platform", y = "Number of Games") +
+   theme(axis.text.x = element_text(angle = 45, hjust = 1))
> ggplot(data = d, aes(x = User_Score)) +
+   geom_histogram(binwidth = 0.5, fill = "blue", color = "black") +
+   labs(title = "Distribution of User Scores", x = "User Score", y = "Count")
Warning message:
Removed 88 rows containing non-finite outside the scale range (`stat_bin()`).
> ggplot(d, aes(x = Critic_Score, y = Global_Sales)) +
+  geom_point(color = "darkred", size = 2, alpha = 0.6) +
+ geom_smooth(method = "lm", color = "black") +
+ labs(title = "Critic Score vs Global Sales by Rating", x = "Critic Score", y = "Global Sales")
+
+ theme_minimal() +
+  facet_wrap(~ Rating)
`geom_smooth()` using formula = 'y ~ x'
Warning messages:
1: Removed 73 rows containing non-finite outside the scale range (`stat_smooth()`).
2: Removed 73 rows containing missing values or values outside the scale range (`geom_point()`).
> ggplot(d, aes(x = Critic_Score, y = Global_Sales)) +
Warning message:
In grid.Call.graphics(C_points, x$x, x$y, x$pch, x$size) :
  semi-transparency is not supported on this device: reported only once per page
+   geom_point(color = "darkred", size = 2, alpha = 0.6) +
+   geom_smooth(method = "lm", color = "black", se = FALSE) +  # se = FALSE to remove the confide
nce interval
+   labs(title = "Critic Score vs Global Sales by Genre and Platform",
+        x = "Critic Score",
+        y = "Global Sales (in millions)") +
+   theme_minimal() +
+   facet_grid(Genre ~ Platform)
`geom_smooth()` using formula = 'y ~ x'
Warning messages:
1: Removed 73 rows containing non-finite outside the scale range (`stat_smooth()`).
2: Removed 73 rows containing missing values or values outside the scale range
(`geom_point()`).
> save.image("C:\\Users\\iswar\\Desktop\\R\\R")
>
```

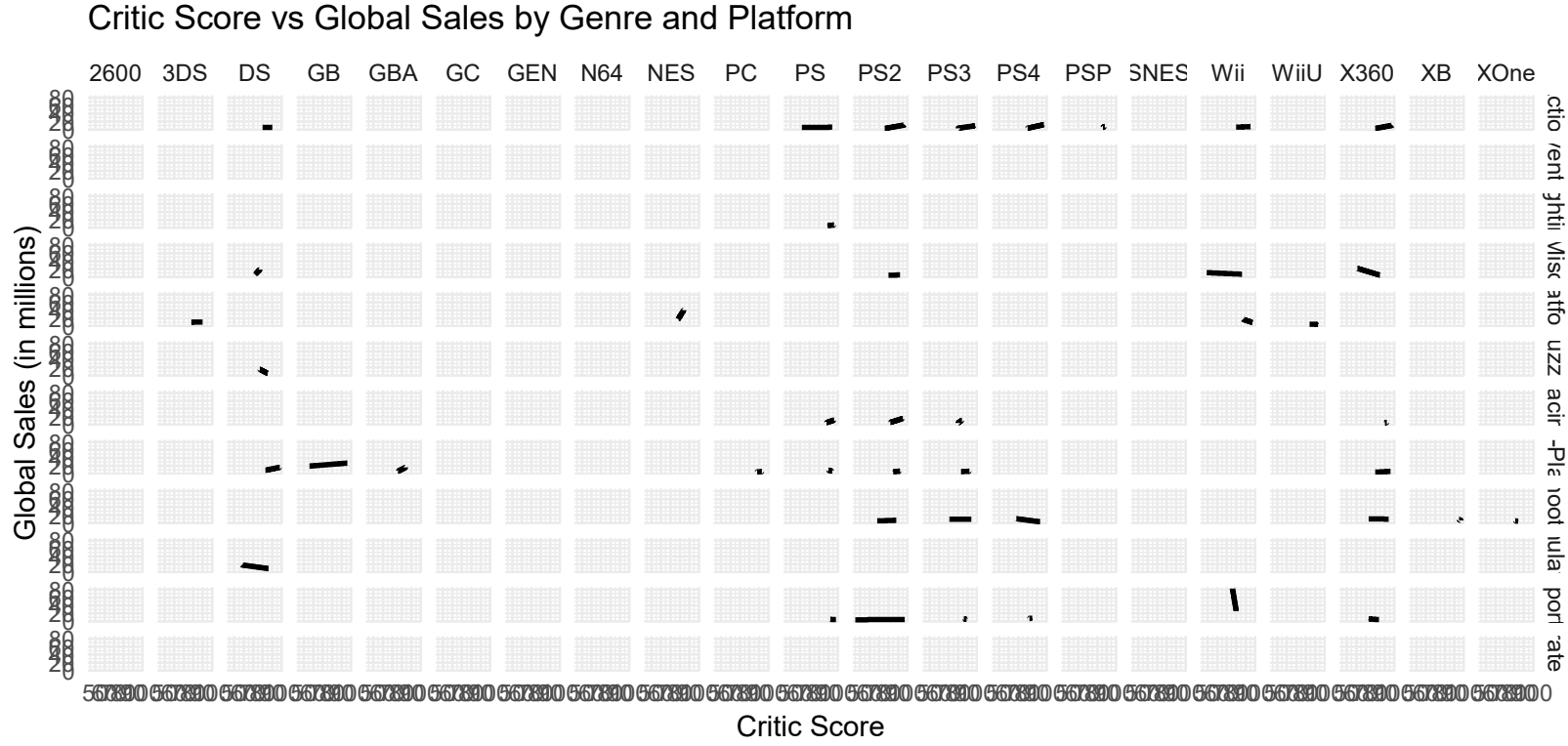

Global Sales vs. Critic Score
Global Sales (millions)
80
60
40
20
50
60
70
80
90
100
Critic Score


Average Global Sales Over Years
Average Global Sales (millions)
40
30
20
10
1980
1990
2000
2010
Year of Release


Number of Games by Platform
Number of Games
40
30
20
10
0
2600
3DS
DS
GB
GBA
GC
GEN
N64
NES
PC
PS
PS2
PS3
PS4
PSP
SNES
Wii
WiiU
X360
XB
XOne
Platform


Distribution of User Scores
Count
50
40
30
20
10
0
0
25
50
75
User Score


Critic Score vs Global Sales by Rating
1
2
3
4
5
Global Sales
0
20
40
60
80
50
60
70
80
90
100
Critic Score


Critic Score vs Global Sales by Genre and Platform
2600
3DS
DS
GB
GBA
GC
GEN
N64
NES
PC
PS
PS2
PS3
PS4
PSP
SNES
Wii
WiiU
X360
XB
XOne
Global Sales (in millions)
Critic Score